UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Marharyta Dekret

SpiralNet: Two-stage recursive-CNN for microscopy image segmentation

Master's Thesis (30 ECTS)

Supervisor: Daniel Majoral, PhD

Tartu 2019

SpiralNet: Two-stage recursive-CNN for microscopy image segmentation

Abstract:
Microscopy image segmentation demands a higher precision level than segmentation for natural images. Meticulous accuracy is required for medical applications. SpiralNet is designed as a new segmentation method allowing to segment microscopy images of complex shapes with high attention to details simulating human perception. The method is able to perform both instance and semantic segmentation. SpiralNet consists of two stages, the first stage crops the initial image into smaller regions and with a scoring network filters out regions without objects. The second stage takes each region and fully segments it with a recursive segmentation network. Afterwards, the predicted regions are merged into the final full prediction mask.

SpiralNet outperforms U-Net with a 0.969 F1 score versus U-Net 0.965 on the test subset, segmenting more accurate individual object shapes and showing better separation between connected objects. Even though SpiralNet showed great instance and semantic segmentation performance, there are still various ways to improve the method. For instance, with parallel segmentation of several regions, adding attention or changing the number of skip modules. Additionally, future work will study the application of SpiralNet to other datasets.

Keywords:
Deep learning, microscopy, segmentation, SpiralNet.

CERCS: P176 - Artificial Intelligence

SpiralNet: kaheetapiline rekursiivne CNN mikroskoobi pildi segmenteerimiseks

Lühikokkuvõte:

Mikroskoopiliste piltide segmenteerimine nõuab suuremat täpsust kui tavaliste piltide segmenteerimine. Ülim täpsus on vajalik meditsiinilisteks kasutusjuhtumiteks. SpiralNet on disainitud kui uus segmenteerimismoodus, mis lubab segmenteerida keerukate kujundite mikroskoopilisi pilte kõrge detailitäpsusega, simuleerides inimese taju. Meetod on võimeline nii üksikobjekti segmenteerimiseks (instance segmentation) kui semantiliseks segmenteerimiseks (semantic segmentation). SpiralNet on kaheastmeline – esimene aste lõikab algse pildi väiksemateks regioonideks ning skooringuvõrguga filtreerib välja objektideta alad. Teine aste võtab iga piirkonna ja segmenteerib selle täielikult korduva segmenteerimisvõrguga. Hiljem prognoositud regioonid ühendatakse lõplikuks täielikuks ennustusmaskiks.

SpiralNet ületab U-Neti taset, saavutades 0.969 F1 skoori U-Neti 0.965 skoori vastu testitud alamkogumis, segmenteerides korrektsemalt individuaalsed objektikujundid ja näidates paremat eristamist seotud objektide vahel. Kuigi SpiralNet näitas nii üksikobjekti kui semantilises segmenteerimises kõrgeid tulemusi, on endiselt mitmeid viise, kuidas seda meetodit parandada. Näiteks oleks võimalik segmenteerida paralleelselt mitmeid regioone, lisada tähelepanu või muuta vahelejäänud moodulite arvu. Lisaks uuritakse tulevases töös SpiralNeti kasutusvõimalusi teiste andmekogumite peal.

Võtmesõnad:

Süvaõppe algoritmid, mikroskoopia, segmenteerimine, SpiralNet.

CERCS: P176 - Tehisintellekt

Contents

1 Introduction

Segmenting microscopy data is very important for medicament research and medical diagnosis. The microscopy segmentation requires high attention to details, because a marginal error can lead to misdiagnoses in clinical settings resulting in colossal harm or even a patient death [Gra13]. The medical images are obtained using different imaging techniques, such as Magnetic Resonance Imaging (MRI), Computed Tomography (CT), Positron Emission Tomography (PET) [KSD+08]. The automatic processing without manual human segmentation reduces error, time and cost [AHY+18]. Thus, there is a high demand for a unified microscopy image segmentation method.

The reconstruction of neural circuits, connectomics, or other cell structures requires the tracing of cells, especially when there are overlapping individual objects or long structures with tiny connections. Therefore, an efficient image segmentation method that can focus on the details of the specific object is highly desirable. The possible applications of such a method are a segmentation for tumor volume measurements during therapy [CVC+95], glioma segmentation in MR images [LGW+18], prostate segmentation in MR images [LGOS13], malignant breast tumors segmentation [RJKK15], knee cartilage segmentation [ZDL11], finding connections between neurons, etc. [AL19].

The current giant of biomedical image segmentation is U-Net [RFB15]. U-Net is trainable on a small number of labeled images what particularly useful for medical segmentation tasks, like tumor detection or cell separation. But the original U-Net is lacking shape precision for medical images and the possibility to segment connected objects individually. Recently, U-Net modifications are developed to improve segmentation accuracy. For instance, a nested U-Net architecture called U-Net++ [ZRSTL18] for medical image segmentation, in U-Net++ encoder and decoder are connected with a series of nested skip connections. However, on selected Nuclei of U2OS cells [dat] the first version of U-Net++ showed quite poor separation capabilities between objects. Thus, U-Net++ was modified by redesigning skip connections that can exploit multidimensional features [ZMTL19]. New U-Net++ significantly outperforms U-Net for semantic segmentation, but for instance segmentation, other approaches, like Mask R-CNN [LL18] should be used. There have been other U-Net modifications aimed to improve segmentation accuracy focusing on target, such as attention U-Net [OSF+18] with attention gate, unfortunately, attention U-Net does not show any performance improvement compared to a baseline U-Net, so the Recurrent Residual CNN based on U-Net [AHY+18] was designed and tested for blood vessel segmentation and skin cancer segmentation. However, the second task does not have accurate shapes what is heavily important for medical purposes. To conclude, there is still no universally accurate medical segmentation method and developing such would be revolutionary for microscopy segmentation.

This thesis aims to deliver a new method allowing to segment low contrast medical microscopy images with high precision and a possibility to distinguish overlapping

individual objects. The whole idea is that the network with the ability to look into the object close and trace this object similarly to the flood-filling network idea in [goo18] should segment better. The existing flood-filling network designed for 3d data, while the majority of medical imaging techniques produce 2d images, additionally, the flood-filling method is highly time and memory hungry. Knowing the weaknesses of existing algorithms a SpiralNet was designed.

SpiralNet consists of two stages. The first stage is for cropping the big image into regions and filtering ones without necessary information in it with the scoring network to reduce time and memory consumption. The second stage takes filtered regions and makes accurate iterative segmenting with the recursive segmenting network. Thus, the scoring network finds regions of interest with an object located in the center and the segmenting network outputs the segmentation for this region. Finally, predicted regions are glued together to their respective places of the final prediction mask.

The thesis consist of the following parts: Section 2 has history of image segmentation, describing basic segmentation terms, such as semantic and instance segmentation. Additionally, it has an overview of related works and existing methods. For semantic segmentation it is U-Net [RFB15] and its modifications: U-Net++ [ZRSTL18], U-Net++ with modified skip connection [ZMTL19], attention U-Net [OSF+18], RCNN based on U-Net [AHY+18], Mask R-CNN [HGDG17] for instance segmentation and Flood-filling networks [goo18] which are the most similar existing method to SpiralNet. Section 3 is describing the architecture of SpiralNet; Section 4 has results, comparison with U-Net and reflections about possible problems and improvements.

2 Related works

This section is a review of what is image segmentation, the historical development of segmentation methods, and current state-of-the-art approaches, such as U-Net and Mask R-CNN. Additionally, flood-filling networks will be reviewed, since it is the most similar method to SpiralNet presented in the literature.

2.1 Introduction to image segmentation

Despite the progress in Computer Vision human perception is still superior. We are able to infer high level abstract features, while algorithms see the picture from the pixel-wise perspective. The field of Computer Vision aim is that the computer understand high-dimensional images and videos. One of the main tasks in Computer Vision is object detection. Typically, object detection pipelines use bounding boxes to locate objects, however, it does not provide any information about the object shape. Object shape is provided by another important Computer Vision task - image segmentation. The idea behind image segmentation is to split image into different segments corresponding to objects. In contrast to object detection, which typically assigns one class to some object region, image segmentation algorithms are assigning a class to every pixel. Each of pixels in this class share the same characteristics and the resulting output contains more meaningful information.

There are two different tasks in image segmentation: Semantic Segmentation and Instance Segmentation. Semantic segmentation assigns to each pixel in an image a category [blo]. The current state-of-art in semantic segmentation is U-Net[RFB15] and it will be discussed in more detail in Section 2.4. Instance segmentation, on top of semantic segmentation divides the image into individual objects. For every pixel it identifies if it belongs to an individual object of some class. The current state-of-art method for instance segmentation is Mask R-CNN [HGDG17] and is discussed in the Section 2.5. Figure 1 illustrates the difference between semantic and instance segmentation which lies in defining objects individually for instance segmentation, while semantic segmentation fuses together objects of the same class.

2.2 Approaches to image segmentation

Medical image segmentation has a long history of progress. Early methods were far from todays deep learning techniques. For example, Roberts, Prewitt and Sobel edge detector operators first time were used and introduced for image segmentation in 1965 according to [Zha1]. Additionally, other methods for segmenting medical images were used for the first time, such as region growing approaches, classifiers, clustering, Markov random field models [PXP00] in 2000. Historically, hand-crafted features [HJHK19] were used as the main approach, however, this method is limited by the problem of extracting

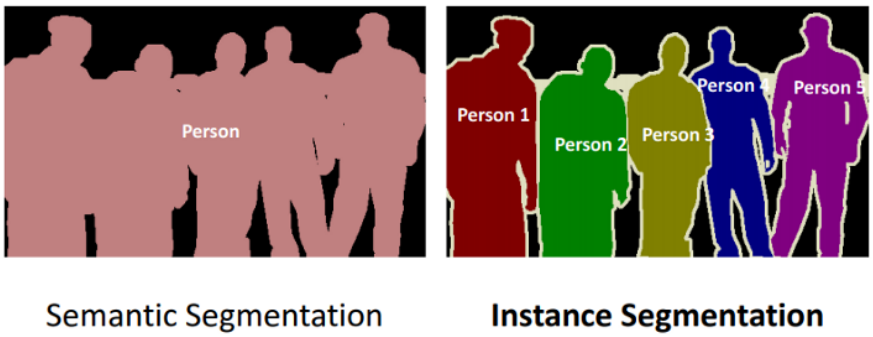


Figure 1. Difference between semantic and instance segmentation [blo]

complex features and design them. Nowadays, with constantly growing computational power deep-learning methods play a dominant role and perhaps are a strong tool in image processing.

2.3 Fully Convolutional Networks

Before 2015, Convolutional neural networks were widely used as classifiers, outputting an individual category label for each image [RFB15]. However, some tasks require pixel-wise localization of classes. From this challenge comes the idea to predict an output for one central pixel in some local region called patch. Using patches as a training data increase the overall number of training samples what is especially useful for medical image segmentation. However, the first approaches consisted in processing patch for every pixel are very slow. Moreover, it is quite difficult to choose the right size of the patch. The smaller regions brings less information and thus, results in smaller accuracy, while the big ones needs more computational power. Long et al. [LSD15] was the first who replaced the last fully connected layer with a fully convolutional layer allowing to predict the network pixel-to-pixel from an original image size. The architecture of Fully Convolutional Network (FCN) is shown in Figure 2. This revolutionary approach outperformed all the methods in image segmentation. Furthermore, Fully Convolutional Network is simple to implement and faster.

Methods and results of the semantic segmentation have been improved quickly after FCN [LSD15] during the last years. Powerfull aproaches were build on the top of the Fully Convolution Network. In the next subsection, we discuss one of the most commonly used methods for semantic segmentation U-Net.

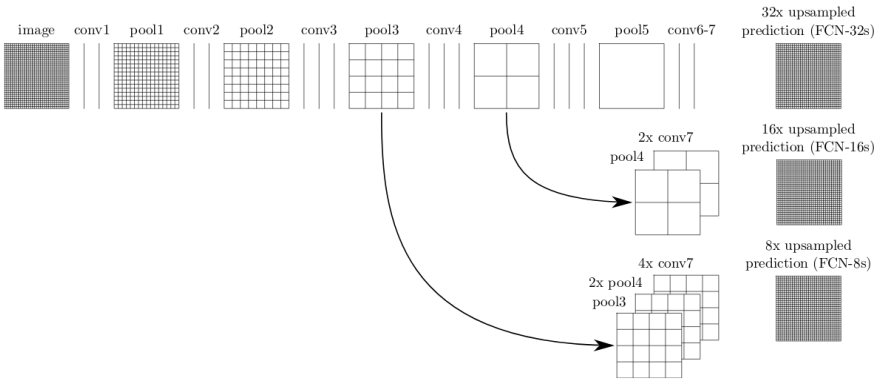


Figure 2. Fully Convolutional Network architecture [LSD15]

2.4 U-Net

U-Net [RFB15] is probably one of the most popular architectures used in medical segmentation. It outperformed all existing segmentation methods in 2015.

The network consists of two paths one is the contracting path and other is the expansive path. Visually these paths folding in a letter 'U' as it shown in Figure 3 from where the name U-Net comes from. The model is an improvement over Fully Convolutional Network (Section 2.3) by using skip connections among the stages in the network. It improves the ability to learn from the smaller number of training images without losing its precision.

The contracting path consists of contraction blocks, where every block has set of 3x3 convolutions followed with ReLU and a 2x2 max pooling. Max pooling decreases the image size (downsampling). And for every maxpool the number of channels is doubled, having the same number of parameters at every step. The reduced image size helps to detect structures of different sizes.

The expansive path is symmetric to the contracting path, but with upsampling at every step. Similarly the amount of channels are halved. The upsampled features are concatenated with the same size features from the contracting path, with use of skip connections. It is a combination of information from previous layers for getting more precise output. At the end, the image is reshaped according to prediction requirements.

One of the advantages of U-Net is an ability to train with just a few labeled images using data augmentation techniques. It is particularly good for medical segmentation tasks when it is usually very difficult to get a big amount of labeled images. U-Net is a simple but powerful model that is still used in all kind of segmentation tasks.

The main problem of the U-Net architecture is the separation between objects when they are close or connected. Particularly in microscopy image segmentation it is often

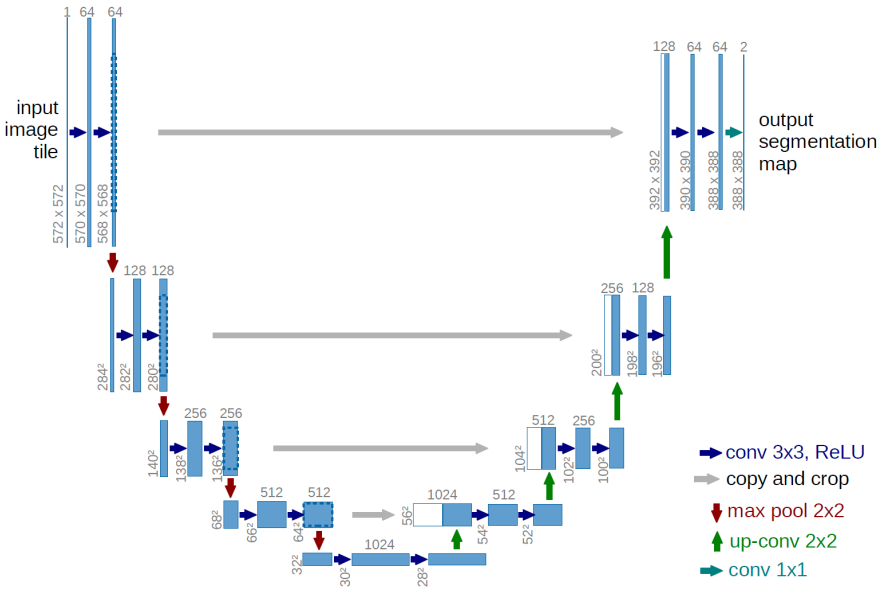


Figure 3. U-Net architecture [RFB15]

crucial to separate neighbour cells accurately. Besides that, U-Net does not perform for instance segmentation. Thus, we describe Mask R-CNN a state-of-art method for instance segmentation in the next subsection.

2.5 Mask R-CNN

Instance segmentation is not an easy task, since it needs a precise detection of all objects while also accurately segmenting every object. Tackling instance segmentation problem usually requires two parts: a precise object detection method and another method to segment the detected object. Despite the new object detection methods developed recently, such as Fast R-CNN [Gir15] and Faster R-CNN [RHGS15], instance segmentation is still an open challenge. It requires both high-resolution segmentation of each object and right detection of these objects.

The current state-of-the-art instance segmentation method was proposed by He et al. in 2017 and called Mask R-CNN [HGDG17]. This method is an extension of Faster R-CNN with segmentation masks added for every detected object in parallel with existing classification and bounding box regression.

First, Mask R-CNN uses an object detection approach from Faster R-CNN which finds a bounding box for each individual entity. Afterwards, Mask R-CNN classifies the individual objects into classes and performs pixel-wise segmentation inside bounding boxes.

The first stage of Mask R-CNN has the same Region Proposal Network as Faster R-CNN use. The Region Proposal Network outputs bounding box, also called as a Region of Interest (RoI). Subsequently, the semantic segmentation part takes that RoI and predicts the object segmentation mask. Mask R-CNN separates class prediction and segmentation mask, which means that without any class competition the binary mask is predicted for each label separately and another branch is predicting the class for this region in parallel. That improves the results comparing to pixel-wise multi-class segmentation with fully convolutional network. Figure 4 illustrates Mask R-CNN architecture.

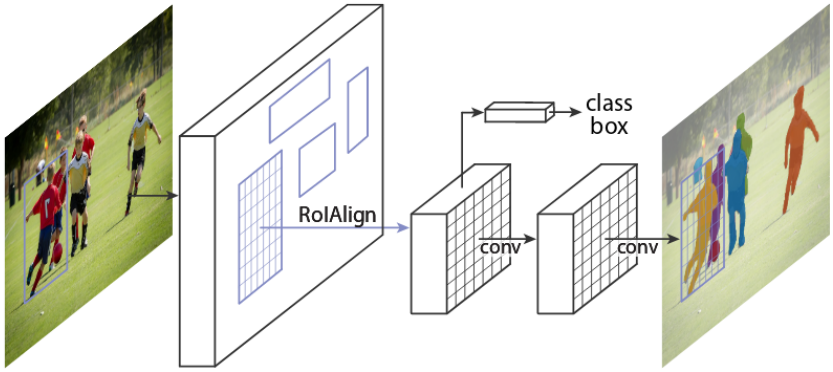


Figure 4. Mask R-CNN architecture [HGDG17]

Unfortunatelly, Faster R-CNN is not constructed for pixel-wise alignment between input and output. Thus, the most tricky part in this method is to apply pixel-to-pixel alignment on the top of Faster R-CNN. The thing is that predicted Faster R-CNN Region of Interest has different shape from the object mask. Moreover, all bounding boxes have different size. For this reason the projection method is constructed. The article [HGDG17] uses RoIAlign algorithm, it is quantization-free layer for projecting all spatial regions to the same size.

Despite the fact that Mask R-CNN is a very powerful method, it has their weaknesses which are particularly crucial for microscopy segmentation. First, it loses some precision because of alignment procedure. Second, there is a problem of extracting objects with rectangles, some objects overlapping each other. Moreover, objects with elongated shapes, what is typical for biological images, such as neurons, connectomics, etc., can not be separated into bounding boxes. Therefore, in the next section we describe flood-filling networks that work better for objects with all kinds of shapes.

2.6 Flood-filling network

Reconstruction of microscopy data requires the tracing of cells, with overlapping individual neighbours or long structures with tiny ramifications. For this reason, the flood-filling approach was developed [goo18]. The method uses convolutional neural networks for 3D volume data with the novelty of a recurrent pathway which iterative predicts pixels. Flood-filling networks [JML+16] increase accuracy by the order of magnitude, but also they increase the computational cost significantly.

The flood-filling architecture takes raw image area and a respecting object mask area as an input to train on stacked convolutional module with skip connection and outputs the object mask. Furthermore, this updated object mask goes as a next input for new iteration (see Figure 5).

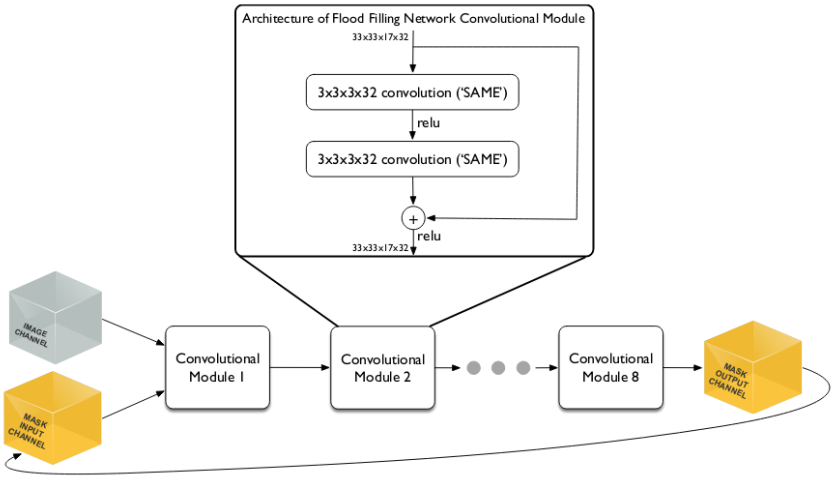


Figure 5. Flood-filling network architecture [JML+16]

The trickiest part in this approach is how to choose the object of interest for the next step, because the method should look for the same object to continue segmenting. For this task the inference procedure is presented, where after each step potential new positions are searched by checking values of current state of the predicted mask. If the value is exceed some threshold, corresponding location will be added to the list of positions which should be visited by the algorithm. When all directions are checked, the location list is sorted and added to a queue of positions. Next, Flood-filling network visits every location from the queue checking all possible directions at each step. At the same time as the queue becomes empty, inference procedure ends.

The idea of the thesis is to apply similar approach for 2D data, since the majority of medical data is in two dimensions. Moreover, we are going to use simpler and

more efficient procedures what will optimize the computational speed with high quality segmentation. Even though, the existing algorithm shows high efficiency, it is computationally expensive and could be improved by removing heuristic inference procedure. Though, simplifying the way of choosing the next area of interest is one of the key difference between SpiralNet and Flood-filling networks. Another point is that Flood-filling method requires seed points what could be avoided if we use a two stage network.

3 Methodology

In this section SpiralNet will be presented. It is designed to segment microscopy images with maximum precision. SpiralNet is able to make both semantic and instance segmentation and outperform current state-of-the-art approaches. Reviewed in Section 2.6 flood-filling approach shares some similarities with this network, however, there is lots of fundamental differences which will be reviewed in this section. The general idea is that SpiralNet can focus closely on one object simulating human perception.

Unlike the flood-filling approach, the proposed SpiralNet will have two stages. First stage has a scoring network, which is designed to select regions with an object in the center. This stage helps to chunk a initial image into smaller pieces with center-located objects and filter unnecessary regions without objects. This procedure allows to decrease memory demands and speed up the second segmenting stage. The second stage has the segmenting network, which will look into a small region and segment the central object. The segmenting network takes patches from the scoring network and produces an object mask as a segmentation prediction of respecting region. The mask from the segmenting network is written to respective areas of the initial image size mask to create the final output.

Figure 6 displays SpiralNet whole pipeline. The initial image is divided into smaller regions, each one of them is marked with a numerical score by the scoring network. After that, crops which pass a threshold are sent to the recursive segmenting network. The segmenting network takes a part of raw image and an empty object mask as initial input and produces a prediction. This prediction updates respective part of the current object mask, which than proceed as a next input with the other part of the image for the second iteration. Iterations continue to cover all parts of the image every time updating the object mask. Thus, the object mask is recursively updated based on previous iteration prediction. Finally, the output is placed to the respective coordinates into the initial size image mask. In the next sections we describe each one of the two stages in more detail.

3.1 I stage: Scoring Network

The main purpose of the fist stage is to look into the big image and decide which part of it goes to the second stage. So, the scoring network is designed for filtering regions of image, so the ones with an object located in the center are left. Figure 7 illustrates how the scoring network crops the image and assigns scores for every patch, after that some threshold could be applied to filter undesired regions.

Approximate region size should be that average object could fit into it, so it is depend on operating dataset. In our experiments, the region size is 52×58 pixels. For the first attempts, regions were taken every 52 pixels without overlapping, but considering that we need all center-located objects cropping with overlapping is needed. Thus, the initial

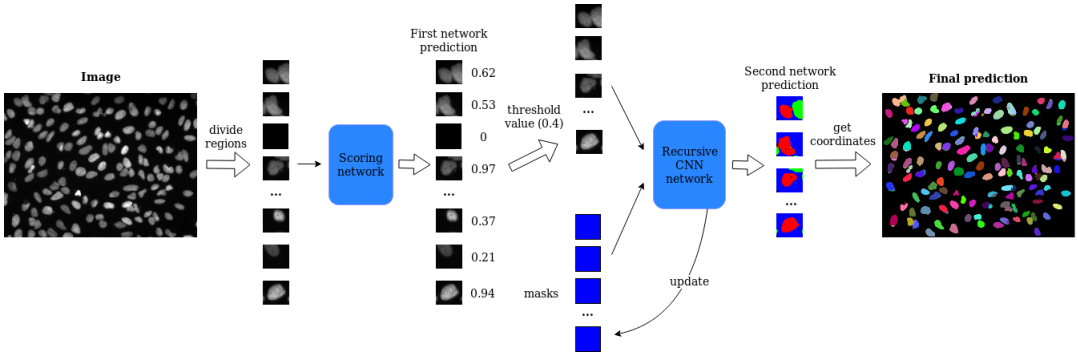


Figure 6. The illustration of the whole process of segmentation from the starting image to the final output.

size image is cropped every 16 pixels of 52×58 from all sides. However, small objects placed on edges could be too small and never reach the center of selected region. To solve this issue, image is padded with 28 pixels from each side of symmetrical image reflection, so the cropping can cover all border objects.

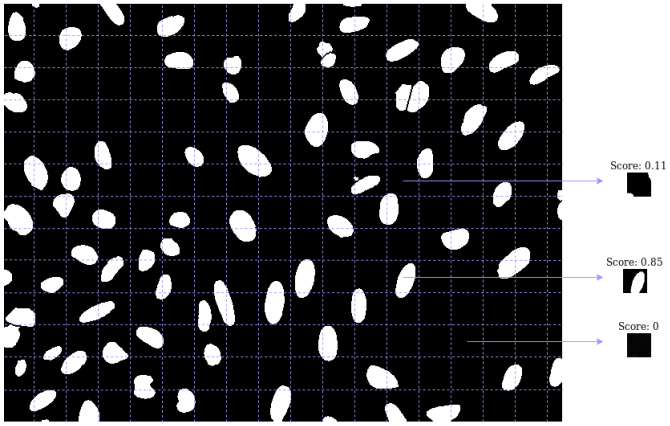


Figure 7. Scores are calculated for every patch of divided image.

The ground truth score is generated in the following way. First, a reverse distance matrix M with the same size as the patch is generated. The reverse distance matrix is zero in the corners and increases linearly until it is one in the central pixel. Afterwards, the ground truth mask is multiplied by the reverse distance matrix. The multiplication with Reverse Distance Matrix is shown in Figure 8. After multiplication the scores are summed up and divided by width and height of the image.

To calculate the score for a concrete patch P with Reverse Distance Matrix M, weight

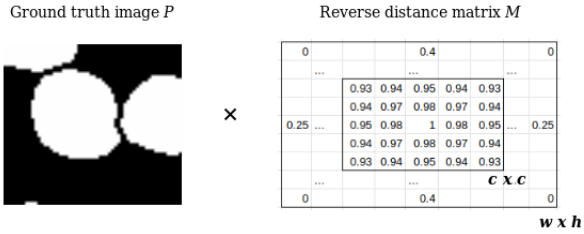


Figure 8. For generating labels image is multiplied by Reverse distance matrix M.

w and height h, the following formula is applied:

$$score = \frac{\sum_{i=0}^{w} \sum_{j=0}^{h} p_{ij} * M_{ij}}{w * h} \tag{1}$$

, where p_{ij} - image pixel on row j and column i. M_{ij} - element of Reverse Distance Matrix on row j and column i.

In practice we obtain better results focusing only on the central area of the patch. Thus, the attention to the central area is needed. The solution is to sum score in smaller central region $c \times c$. Thus, the score formula (1) could be modified as follows:

$$score = \frac{\sum_{i=0}^{c} \sum_{j=0}^{c} p_{ij} * M_{ij}}{c^2} \tag{2}$$

, where $c \leq w$ and $c \leq h$ - is a parameter for finding score in central region. For the results presented here we have used the value $c = 5$ and the final score calculation comes from multiplying a 5×5 area in the center with 5×5 part in Reverse distance matrix and dividing by 5^2 (formula (2)). Some generated label scores after this procedure are shown in Figure 9. The aim for the scoring network is to learn these scores from the raw image, so regions without object located in the center could be filtered out.

After finding label scores for every region the scoring network should learn to predict scores from raw images. For building the network approach similar to spatial attention model in [LXPS17] is used. It is two-layer linear model with Mean Absolute Error (MSE) as a loss function. The input image channel is flattened and goes through one linear layer with 128 nodes which than goes through tangent and softmax functions, after that the second linear layer is applied with 1 neuron to get the single output score.

Finally, the prediction scores are filtered depending on a threshold t, the regions with score higher than t should have object located in the center, otherwise the region is discarded. Empirically, threshold has been set to 0.4 as can be seen in Figure 10.

As could be noticed from the Figure 7 there are cells that are not covered with single division. But making overlapping crops on the prediction stage solves this issue and at the same time it does not increase training time and memory consumption. The network is resilient to changes of input size.

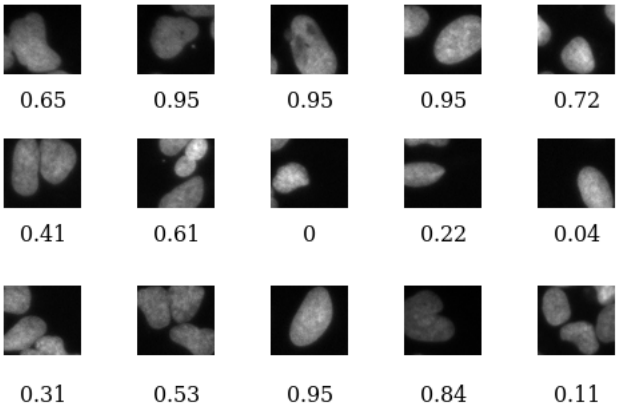


Figure 9. Generated label scores for scoring network.

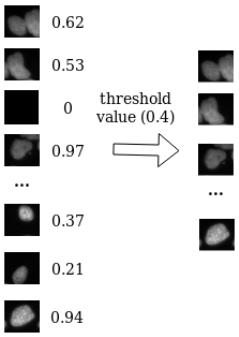


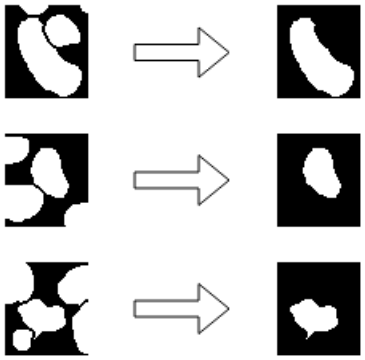
Figure 10. Example of images passing the threshold.

Once image regions are filtered with a threshold the second stage takes place.

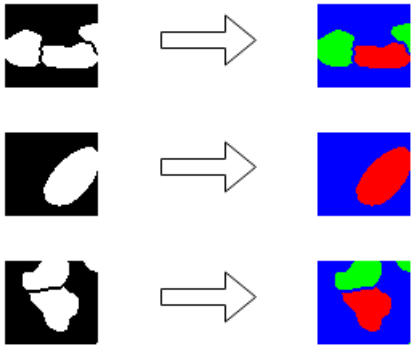
3.2 II stage: Recurrent CNN Network

In this stage we need to segment the central object using first stage outputted regions. However, on cropped patches there could exist parts of other objects (located not in the center), however only the central object must be segmented. For this reason labels transformation is needed. There are two strategies to transform labels so the network can learn to separate the central object from others. First, labels go through the filtering procedure, when only the central object is left, while all others are filtered out (Figure 11a). Second, a label will have 3 channels, where the first channel has ones on the central object and zeros everywhere else, second channel has ones on all other cells except the central one and the third channel has ones for a background only (Figure 11b). Figure 11 shows both strategies. At the beginning of experiments the first scheme were used for

label transformation, but at some point 3-channel option showed to be more efficient, so the results presented here use the second method.



(a) Filtering labels with leaving only central located object and throwing other objects out.



(b) Creating three channel labels: first for central object, second for other objects and third for background.

Figure 11. Two options of label transformation procedure.

The recursive segmenting network has three inputs: raw image regions, their respecting labels and last predicted object masks. Image regions and their labels come from the scoring network prediction, while object mask is initially a zero matrix of the same size as the label patch. Input structure of the segmenting network is shown in Figure 12.

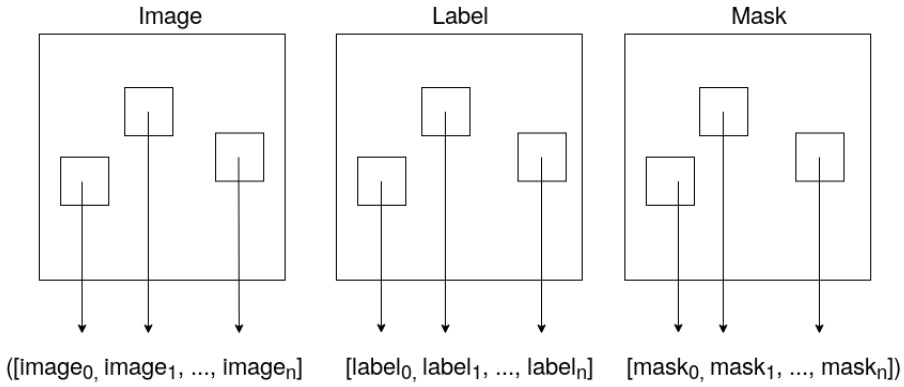


Figure 12. Input which goes to the recursive segmenting network. There are image regions that pass the scoring network threshold, their respecting transformed labels and the object zero mask.

The first iteration of the recursive segmentation takes a field of view of a raw cropped image and a zero object mask as input and predicts segmented output. After that, the

output updates the respective part of the object mask which becomes the next input of the second iteration. The field of view of the first iteration segmentation starts from the center and unlike Flood-filling network [JML+16] the next area to be segmented is chosen regardless the output on a previous iteration. The field of view for every iteration of segmenting network is a region of $m \times m$ size. The recursive network moves spirally starting from the center, afterwards n pixels up, then left counterclockwise and so for until the last corner. The detailed illustration of moving could be seen in Figure 13. After every iteration the network updates an $n \times n$ area in the object mask before the next iteration. For taking $m \times m$ field of view with respect to borders both original image and object mask are padded with 0 of $\frac{m}{2}$ width from all sides. This allows to make predictions area for border pixels.

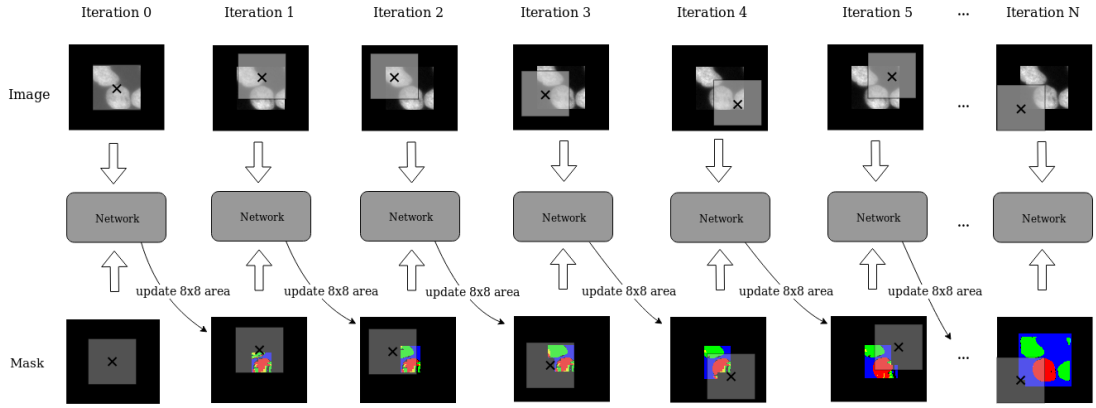


Figure 13. Iterative filling the region by spiral moving starting from the central area all way until ends at the corner.

The segmentation network is illustrated in Figure 14. The network starts with a padded original image as input of size $3 \times 104 \times 116$, where $3 \times 52 \times 58$ is the original image and the rest is filled with 0. Next, the current observed area is cropped to the size of $3 \times 52 \times 58$ and goes to the convolutional layer with 3×3 kernel followed by ReLU. The same exact sequence is applied for an object mask which is also taken as an input. Afterwards, mask and image are concatenated together into $64 \times 52 \times 58$ and proceed through the next 3×3 convolution followed by ReLU. After that, the stack of skip modules is applied (Figure 15). Finally, a 1×1 convolution with ReLU is applied to get the $3 \times 52 \times 58$ output and after the sigmoid the prediction is ready for updating current mask before the next iteration.

The skip module could be seen in Figure 15. It consists of two convolutional layers with skip connection between them, similar to [JML+16], however our skip module has two layers with 32 and 64 numbers of neurons, which proved to work better. The number of skip modules can be changed depending on the dataset. Empirically, two skip modules

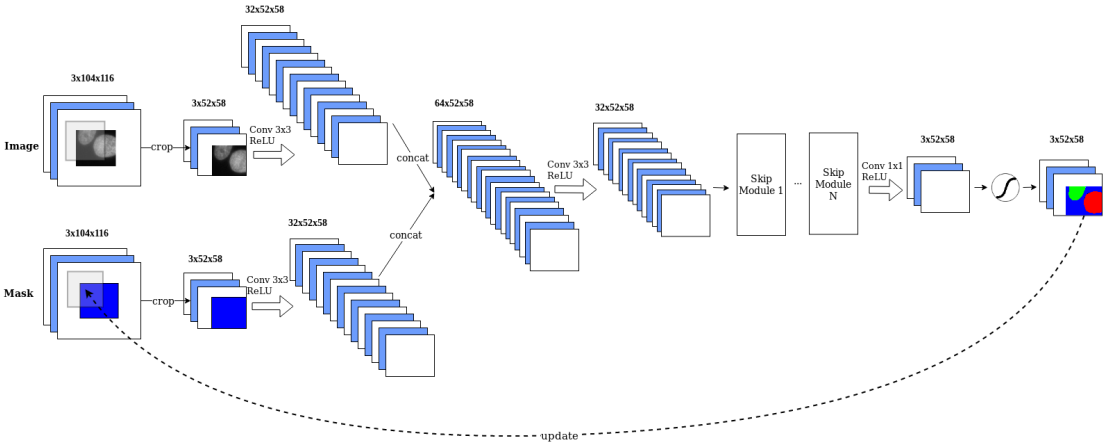


Figure 14. Recursive CNN Network architecture.

was optimal for the dataset employed.

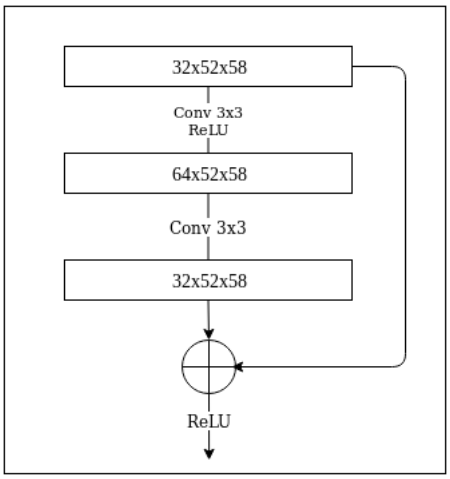


Figure 15. Skip module of recursive segmenting network.

There are objects that go outside the selected region. This situation is solved by keep segmenting when there are central cells prediction on the border. For example, if the right border of the region is segmented we move to the region on the right and apply the segmentation network there. Practically, after getting the prediction all borders are checked if the central object is on it or not. If the object is continuous to the border, the segmentation spreads in direction of that edge. We should ensure that next region will have the same object in the center of the next region, so the center of the object on the

edge becomes the central coordinates of new region and other coordinates are moved respectively (see Figure 16).

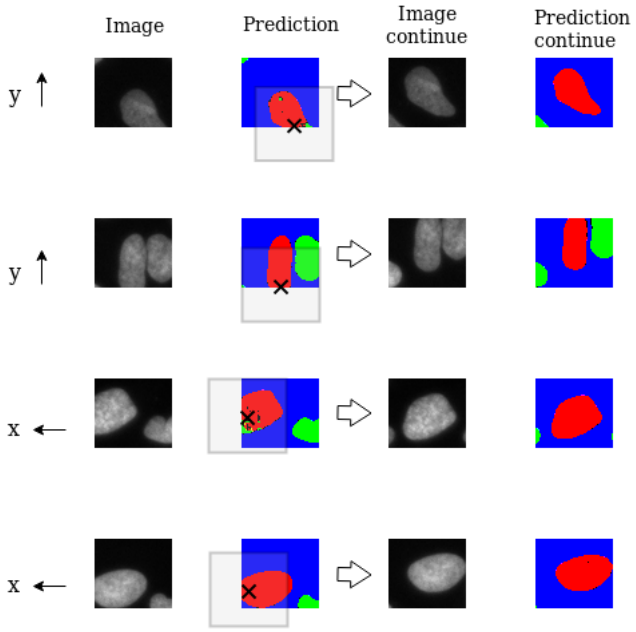


Figure 16. Continuous prediction if the object is not fully segmented.

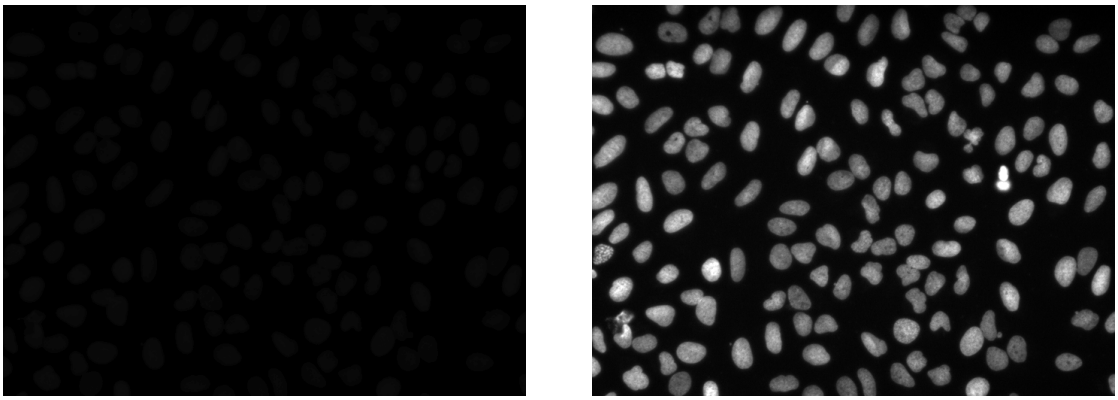
To summarize SpiralNet, the initial size image is cropped for the smaller regions, which goes through the scoring network. Next, they filtered out with threshold, so regions with object located in the center are left. Further, these patches together with masks of zeros go to the recursive segmenting network which takes object mask and image as an input and based on the prediction updates the respective area of view in the object mask before the next iteration. After updating every pixel in one region the network checks if the object is not finished on the edge. If the object lasts, the center is moved to the border of the predicted object and the recursive network will continue segmenting. When the object is fully segmented the prediction is stored in its respective place.

All networks were trained on University of Tartu high performance cluster using NVIDIA Tesla V100 GPUs with 16 GB of VRAM. The code was written using PyTorch framework.

3.3 Data description

For all experiments a dataset consisting of Nuclei of U2OS cells in a chemical screen [dat] was used. It is a collection of around 23,000 single nuclei manually annotated in 200 images of 520x696 pixels.

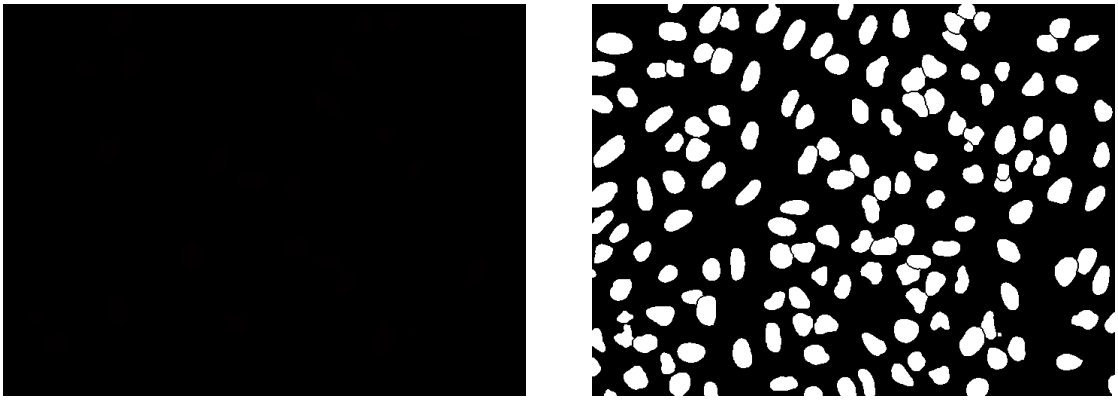
The original pictures are low contrast (see Figure 17a), so the min-max normalization procedure has been applied. An example of normalized image is shown in Figure 17b.



(a) Raw image before normalization. (b) Raw image after normalization.

Figure 17. Example of raw image before and after modification.

The annotated ground truth images (Figure 18a) have background marked as 0 and cells as 1. However, if two cells touch, they are labeled with different numbers depending on how much cells are connected. First, the ground truth was decoded and transformed to a single channel. Additionally, all cells were label as the same number for generalization of the training procedure. The transformed ground truth is shown in Figure 18b.



(a) Raw annotations before transformation. (b) Ground truth after transformation.

Figure 18. Ground truth example before and after modification.

4 Results

On selected Nuclei of U2OS cells dataset a series of experiments were performed to train and evaluate SpiralNet. First, the scoring network from the first stage was trained for defining center-oriented object regions. Next, the segmenting network was trained with different sets of parameters. Finally, the comparison between SpiralNet segmentation and U-Net was made. In the last subsection 4.6 there will be described other approaches and parameters I have tested, but they were not successful. However, those experiments still might be useful for the future research.

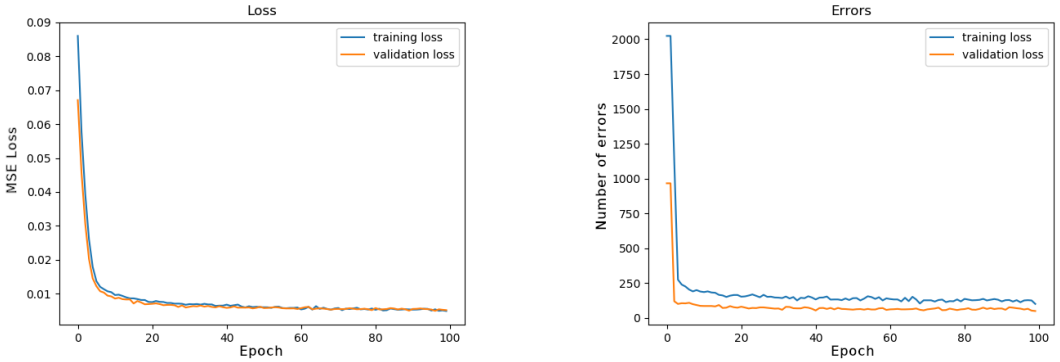
Before training, Nuclei of U2OS dataset was preprocessed. Raw images were normalized with min max normalization and the dateset was split to train (100 images), validation (50 images) and test set (remaining 50 images).

4.1 Scoring network training

The scoring network was trained using Adam optimizer with 0.0001 learning rate for 100 epochs. The Figure 19a shows MSE loss within epochs for training and validation set. Both MSE loss and number of errors graphs were falling drastically at the beginning and gradually decreasing for later epochs (see Figure 19b). The number of errors here is regions which have score passing the threshold 0.4, while there is no object in the central pixel. For the last epoch the scoring network has 7.56% error rate for training subset and 6.02% error rate for validation subset (error rate is number of errors divided by number of selected images and multiplied by 100%). However, this error rate does not represent the network performance, it reveals the problem of generated scores. Because the majority of errors are for cells that are one pixel away from the center and the generated ground truth scores also passes the threshold, while there is no object in the central pixel (see Figure 21).

After training, we can compare the ground truth scores and the scores predicted from raw images (see Figure 20). For objects located clearly in the center the score is usually above 0.9, but cells which are located in left or right side of region have scores lower in the $0.4 - 0.9$ range, objects that do not fill central area are in the $0 - 0.4$ score range. The scores predicted have the same pattern as the labels generated, so we can conclude based on error rate and prediction that the network learned to recognize the central cell.

The threshold selected for filtering affects the second stage performance significantly. From one side, a lower threshold adds more information for next stage training, but on the other side it confuses the segmenting network. To understand how the threshold affects performance a new error rate was computed. The number of regions that passes the threshold for predicted scores is compared with the amount of regions that passes the threshold for ground truth scores. The difference between these two amounts is divided by the number of all regions and multiplied with 100%, the resulted number we called



(a) MSE loss for the scoring network.

(b) Number of errors per epoch.

Figure 19. Loss and errors graphs for the scoring network

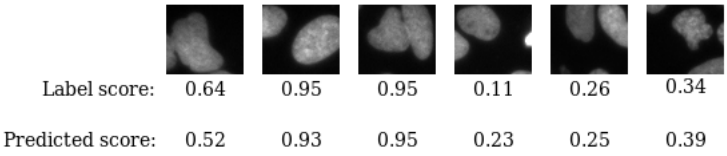


Figure 20. Examples of scores after training.

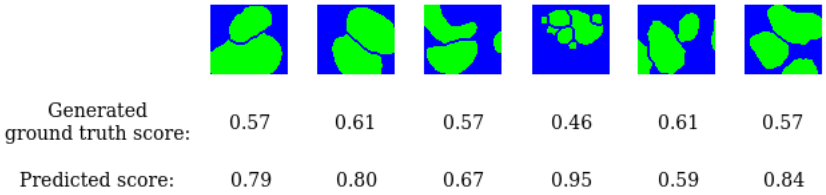
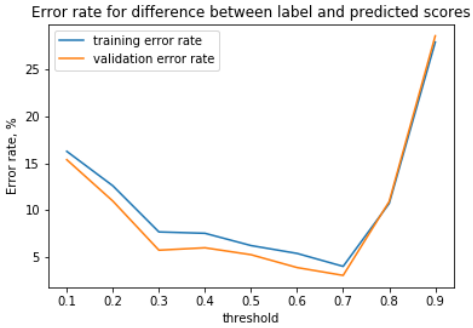


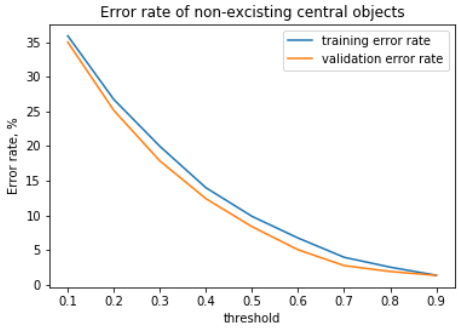
Figure 21. Examples of scores that pass the threshold 0.4, however, they don't have object on the central pixel, because the object located one pixel away.

the stability error rate. Thus, for different thresholds the stability error rate is presented in Figure 22a. Figure 22a shows that the smoothest area is $0.3 - 0.7$, so these thresholds are stable enough for consideration. As expected, the number of misclassified regions that do not have a central object in it with bigger threshold is decreasing (Figure 22b), however, in the $0.7 - 0.9$ range this tendency plateaus.

Considering Figure 22a, 22b thresholds $0.3, 0.4, 0.5$ were tried, threshold 0.5 showed good performance, however, problematic cases confuses the network especially with connected objects, because it did not see enough cases in the training data. On the other hand, threshold 0.3 increases the number of selected regions significantly, but the loss for



(a) Error rate represents the difference between amount of images passing threshold for predicted scores and amount of images passing score with labels.



(b) Error rate, where number of errors represents how much images pass threshold meaning that they have object located in the center, but they do not.

Figure 22. Error rates for different thresholds

training and validation fluctuated a lot and the training process did not converge. Finally, threshold 0.4 showed good results and was selected for further experiments.

4.2 Segmenting network training

Once image regions surpass the scoring network threshold in the first stage, the training labels should be transformed before the second stage starts. The transformation would allow segment only the central object and ignore the others as described in Section 3.2. To characterize the central object we search for it with a connected components algorithm. We employ a three channel label: if a connected component is in the region center, it is written to the first channel, all others objects are written to the second channel and background is written as ones to the third channel (see Figure 11a in Section 3.2).

The network takes an area of raw image and object mask as an input to generate a prediction. Next, this prediction updates a respective part of the object mask before the next iteration. During experiments we tried different area sizes and and number of updated pixels per iteration. On the one hand, the bigger area network could see, the better it could possibly segment. On the other hand, a bigger image each iteration implies higher memory consumption. It is possible to update all the predicted pixels for the whole region size 52×58 pixels with one iteration. However, experiments showed that rewriting the whole object mask area does not give precise segmentation. While at the same time updating only one the central pixel of the region spirally would require $52 \times 58 = 3016$ iterations for every epoch and an extremely long training time. The more pixels per iteration are updated, the bigger step network could take to move from one pixel location to another and reduce the overall number of iterations. To address this

issue, a sequence of experiments was performed with different sizes of the prediction update. Good results were obtained with an 8 pixel updating size. So, every time the network updates an 8×8 area in the object mask and for this 48 iterations per epoch are needed. Figure 23 illustrates the process of how network takes its input and updates object mask is shown.

Finally, the segmenting network was trained for 100 epochs with Binary Cross-Entropy loss and Adam optimizer (0.0001 learning rate). The loss for training and validation set is shown in Figure 24. Both training and validation loss fluctuated a lot in the beginning, but afterwards diminishes smoothly until the 92th epoch when the validation loss reaches its minimum and starts to overfit slightly.

For both training and validation subset F1-score, precision and recall are presented in Table 1. There are three classes: the central object (first channel), other objects (second channel) and background (third channel).

Table 1. Evaluation metrics for segmenting network.

Metrics	**Central object**			**Other objects**			**Background**		
	training	validation	test	training	validation	test	training	validation	test
F1 score	0.946	0.945	0.945	0.893	0.890	0.892	0.972	0.983	0.985
Precision	0.960	0.961	0.962	0.928	0.930	0.924	0.987	0.985	0.987
Recall	0.950	0.946	0.945	0.896	0.890	0.897	0.969	0.981	0.982

The most important here is the central object class, because only the central object will be written to the final prediction. For both training and validation set precision is better than recall, so mostly the segmented pixels are correct, but misses some object pixels. But the visualization of the results reveals very specific cases, which we discuss in Section 4.5.

Example of regions segmented by the recursive segmenting network are presented in Figure 25. This figure has examples where the object is not finished and SpiralNet keeps segmenting by moving a region when the object is beyond the border. The first row of Figure 25 predicts pixels on the right edge, so the next region moves to the right with x coordinate. The new center is the middle of the object border and the segmenting network knows that it is the same object. Similarly, the 5th row predicts the object on the bottom border and it moves half image down to keep segmenting. The other examples have full objects inside one region with successful segmentation.

4.3 Final prediction

When both networks are trained, final prediction could be generated for train, validation and test sets placing individual predictions from the second network to its corresponding coordinates of the full final prediction. The segmenting network outputs three classes:

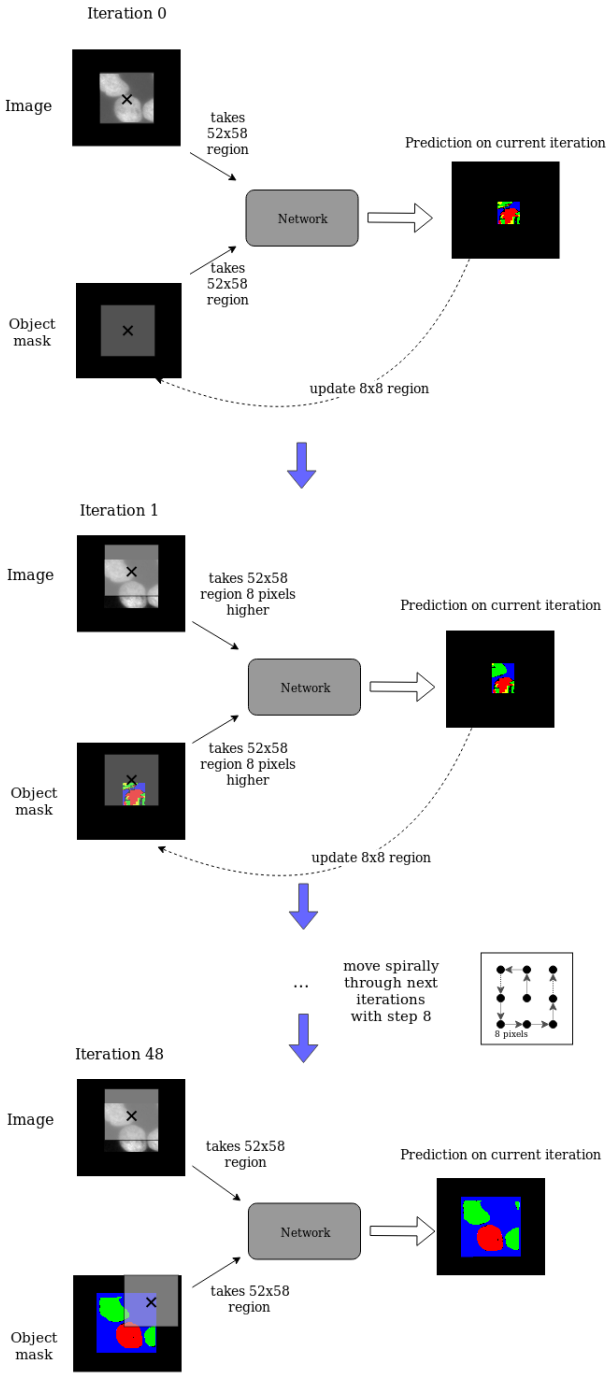


Figure 23. Visualization of area of view movement and prediction updating in the object mask for SpiralNet.

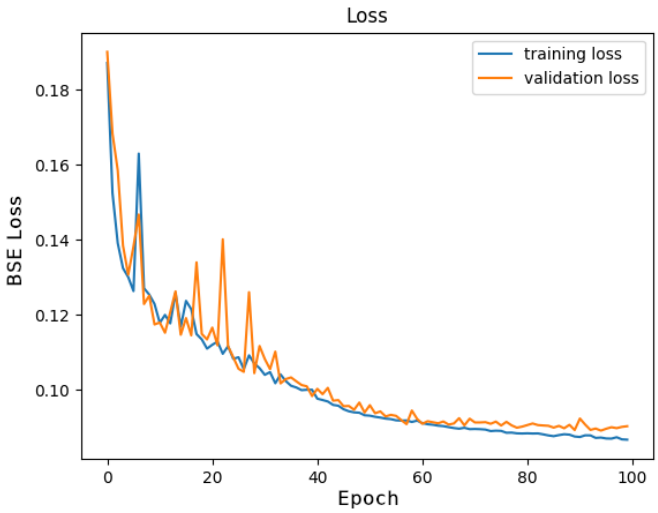


Figure 24. Training and validation Binary Cross-Entropy Loss for segmenting network.

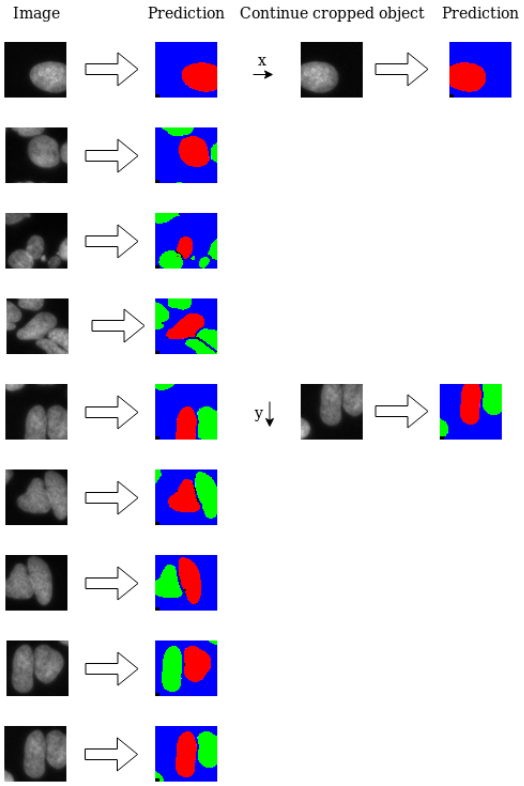
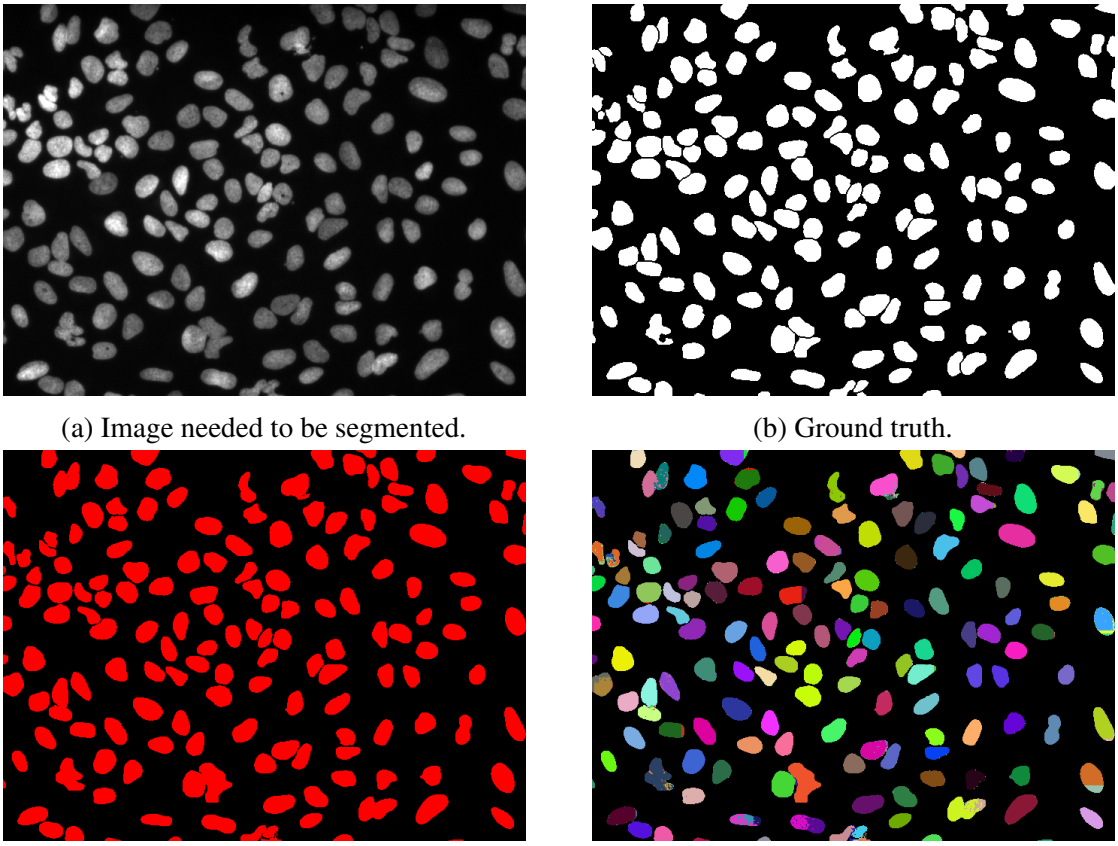


Figure 25. Examples of segmented regions.

central cell, other cells and background. However only the central object is written to the full final prediction. For semantic segmentation all central objects are written as one class, while for instance segmentation each region's central cell prediction will have a different individual number.

Except cases when prediction is not ended with one regions and network moves to continue segmenting the same object, than all regions belong to this object will get the same number. Figure 26 showcasaes a prediction for a test set image network performance. The image shown has lots of objects and nicely illustrates network performance, it contains a considerable number of cells different sizes and almost all of them are well separated with SpiralNet. However, there are few problematic cases when connected cells have the same contrast and are hardly separated even with human eye. The instance segmentation (Figure 26d) shows in colors each individual object, overall it looks good, but could be improved in future work.



(a) Image needed to be segmented.

(b) Ground truth.

(c) Semantic segmention result.

(d) Instance segmentation result.

Figure 26. Example of test image predicted for semantic and instance segmentation.

Generated images are evaluated using the same metrics: F1 score, precision and recall

(Table 2). Surprisingly, validation numbers are higher than for training, but after looking into original images was revealed that in training set presents some problematic cases, which will be discussed in Section 4.5. Overall, scores are very high and comparable with existing well-known segmentation networks. So, in the next section a comparison with U-Net is performed.

Table 2. Metrics for SpiralNet final prediction.

SpiralNet	Training set	Validation set	Test set
F1 score	0.963	0.965	0.969
Precision	0.956	0.960	0.963
Recall	0.971	0.971	0.977

The metrics in Tables 1, 2 compare SpiralNet with the Dataset ground truth. However, as shown in Figure 27 in some cases SpiralNet segmentation is better than the ground truth, but it is penalized for it in the metric values. For instance, in region 1 (Figure 27) SpiralNet prediction covers a small point on top of the biggest cell. Also in Figure 27 in the region 2 some cells have small holes, although they are clearly visible in the original image and predicted correctly by SpiralNet, they are not present in the ground truth labels.

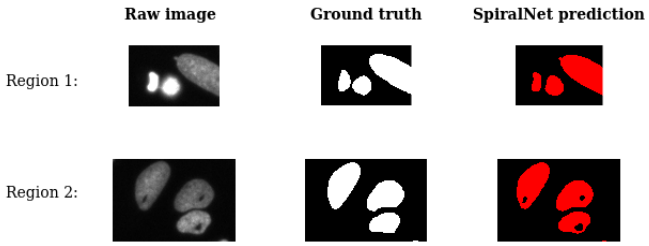


Figure 27. Regions where SpiralNet is more precise than ground truth

4.4 Comparison with U-Net for semantic segmentation

To prove that SpiralNet could be competitive with current state-of-the-art methods, U-Net was trained on the same dataset. Unlike SpiralNet, U-Net can only do semantic segmentation, so the comparison will be only for it.

U-Net was trained for 100 epochs on Binary Cross-Entropy loss using Adam optimizer with 0.00001 learning rate. Figure 28 illustrates the training process for U-net for training and validation subsets.

Figure 29 shows the difference between U-Net and SpiralNet prediction on test set. For U-Net segmentation objects placed close to each other does not have a clear

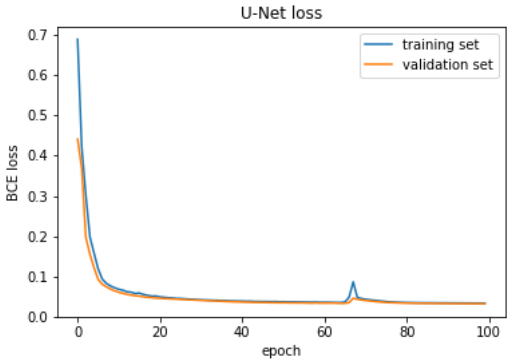


Figure 28. Training and validation loss for U-Net.

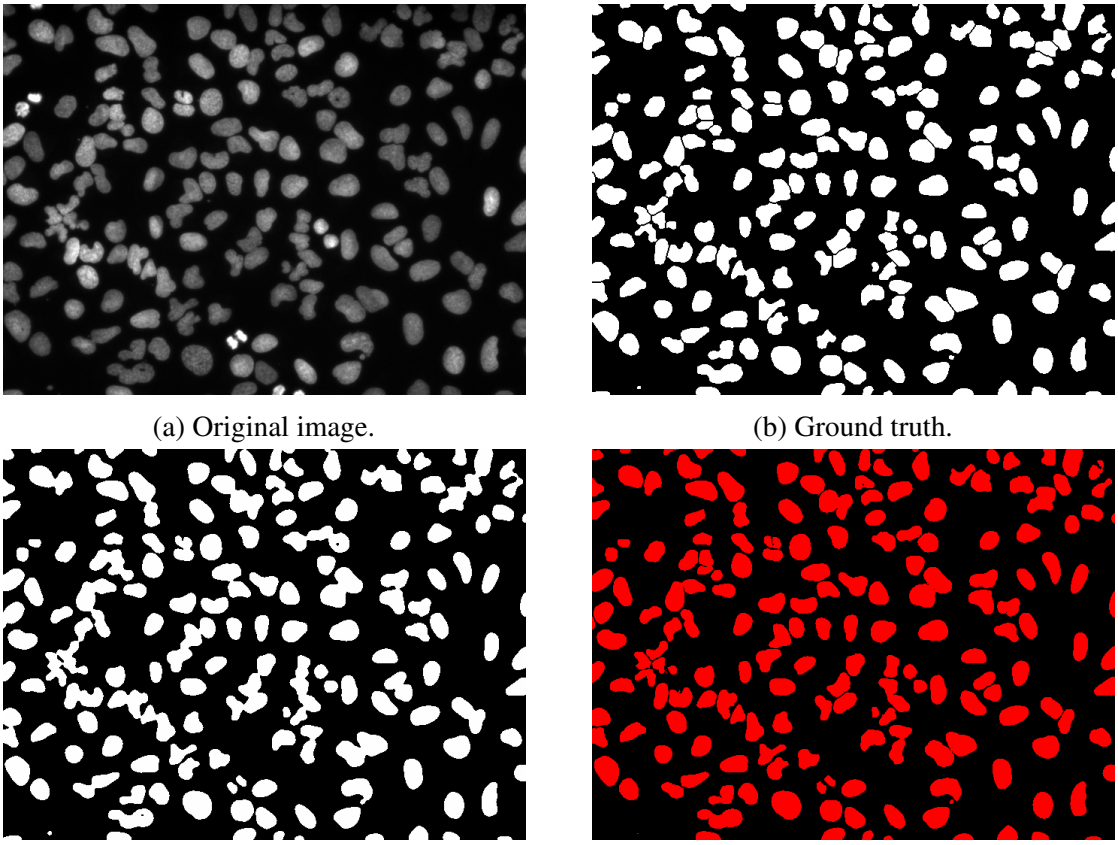
separation and are displayed as one object (Figure 29c), while SpiralNet separates clearly even for connected objects (Figure 29d). However, SpiralNet could have small objects not covered with segmentation, because during division procedure these objects did not get into center of any region. But these errors located in the borders and not really significant, and can be solved with a dense division in the prediction procedure.

Figure 29 shows that SpiralNet separates connected objects more clear than U-Net. Additionally, individual shapes seem to be more precise. This result correlates with expectations that the network iterative process at a very small scale is more accurate, the presented results corroborate this hypothesis. For a more precise evaluation a metric comparison was performed. Table 3 presents F1 score, precision and recall for training and validation set.

Table 3. Metric comparison between U-Net and SpiralNet.

Metrics	**U-Net**			**SpiralNet**		
	training	validation	test	training	validation	test
F1 score	0.963	0.960	0.965	0.963	0.965	0.969
Precision	0.963	0.961	0.969	0.956	0.960	0.963
Recall	0.962	0.959	0.961	0.971	0.971	0.977

From Table 3 SpiralNet has slightly better overall performance for all subsets. But with knowing that SpiralNet tend to miss very small objects and this metrics could be improved by doing more dense prediction and by tuning hyperparameters, and sometimes SpiralNet is able to segment small artifacts from original image better than labeled ground truth (see Figure 27) what put the metrics score down, so we can conclude that SpiralNet outperformed U-Net.



(a) Original image. (b) Ground truth.

(c) Segmenting with U-Net. (d) SpiralNet segmentation.

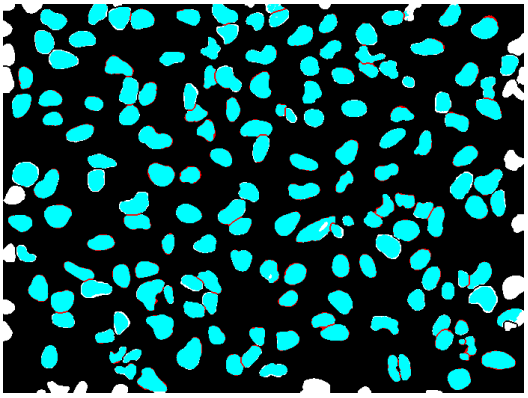
Figure 29. Example of test set image segmented with U-Net and SpiralNet

4.5 SpiralNet weaknesses and challenges

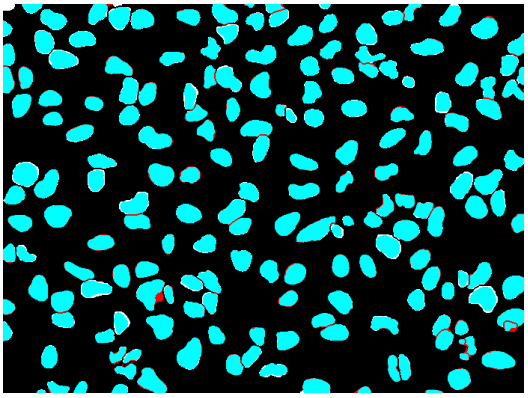
Even though SpiralNet easily outperformed U-Net it still has lots of thing that can be possibly improved. In Section 4.6 some things that were tried so far for improvement will be discussed. But in this section we reflect about current weaknesses and challenges.

There are ways to improve division and filtering procedure, so that it will not be possible to miss smaller objects or the one located in the border. At first, when the network prediction was generated, a border problem had been revealed in Figure 30a. This figure has the difference for prediction and ground truth. After seeing this, the solution to make image reflection on the borders has been applied and the new difference between prediction and ground truth does not have missing objects anymore (Figure 30b), however if we find possible way to avoid such algorithmic complications on a previous stage, we should do it.

The procedure of generating ground truth scores should be improved, as Figure 21 showed, when the object is one pixel away from the center the region should not pass



(a) Difference between SpiralNet prediction and ground truth when border issue is not fixed yet.



(b) Difference when the border issue is fixed with padding. The solution is to make symmetrical reflection for borders.

Figure 30. The difference between SpiralNet prediction and ground truth. White color is represent those pixels that are not predicted with SpiralNet and red color shows false positives.

the threshold, but it does. Thus, it might be worth to look only into the central pixel instead some central region or make the differences in Reverse Distance Matrix more perceptible.

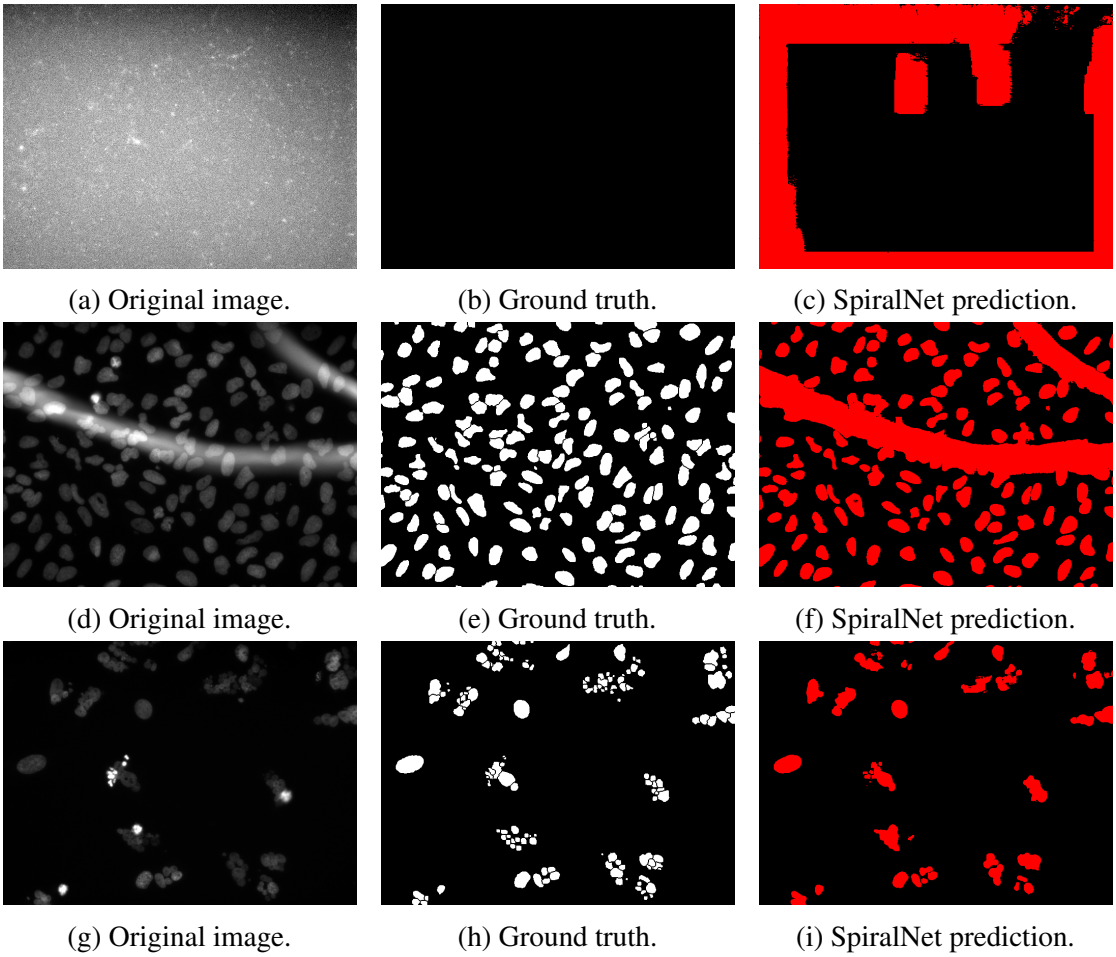
Other challenge is when the image is noisy or have some distortions somewhere, for example in Figure 31. Those examples are in training dataset and cause an important decreasing for metric scores in Table 2, because of that validation set metrics are bigger than training set metrics. Figure 31a is an example of complete noisy image and the prediction (Figure 31c) is unable to ignore the noise. Figure 31d presents some artifacts and the netwrok marks them as cells. Image 31g has very tiny low contrast objects, the network (Figure 31i) fails to separated this kind of objects.

Even though metrics and outputted segmentation are good enough, changing hyperparameters and trying more skip modules could improve the network capabilities. Summarizing all mentioned above, the architecture can be further improved, but even now the network preforms quite good for medical segmentation and has high potential.

4.6 Other things I have tried

On the way of building SpiralNet a sequence of modifications was tried. Some of them failed, but all they helped to learn how to move in the right direction. Here is the list of things which were tried and possible changes that can be applied in the future.

- One of the idea was to add ResNet pretrained weights in the beginning of segmenting network. However, pretrained module increased overfitting and results were



(a) Original image. (b) Ground truth. (c) SpiralNet prediction.

(d) Original image. (e) Ground truth. (f) SpiralNet prediction.

(g) Original image. (h) Ground truth. (i) SpiralNet prediction.

Figure 31. Example of noisy images in training set, where SpiralNet struggles to make correct prediction.

not worth trying.

- One of the crucial steps in the segmentation network architecture is taking different parts of an image during spiral moving (see Figure 13, 23). When cropping far away from the center of the region zero padding was employed. But instead, it could be padded with real data from the initial big image. Beyond taking more memory per iteration to train, it is still a bad idea. Because the bigger area presents more objects that are not located in the center, what lead to imbalance, the network predicts always other object class without the central object class. Although, we still believe that other parameters could improve the situation and gives a valuable result, and we can explore more this hypothesis in the future.

- Initially to predict the central object a different label generation procedure was

used. Consisted of leaving only the central cell and not predicting the other cells class completely (see Section 3.2 and Figure 11a). Switching to three channels labeling (Figure 11b) improved the network performance significantly.

- We added Batch Normalizarion to the segmenting network inside the skip module. Unfortunately, results become worse and the network was not able to learn cell segmentation at all.

- Among others experiments in skip module architecture, the number of these modules was also changed. The set of $1, 2, 3, 4, 8, 16$ number of modules was tried for the segmenting network. The best results were achieved with 2 and 4 modules depending on the learning rate, number of epochs and batch size. But after those experiments labels were switched to three channels , so the information for final experiment might be outdated. Nevertheless the final architecture with 2 skip modules works well although changing the number of modules could possibly improve results.

- Another parameters which were varied a lot are the field of view and how many pixels to update. At the beginning the field of view was the full region size and it was updating only the central pixel. Also seeing and updating the whole area was tried, but it did not work as good as a limited update. Next, we changed both the viewed and updated area to 16×16, 8×8 and 4×4. The smaller area was able to separate connected objects better, however, it predicts sometimes part of the central cell as other cells, because there is not enough information to understand the shape. So, finally the best solution was to have a big field of view and a small updated area. However the more pixels we update per iteration in the object mask, the bigger step in the spiral moving, implying less number of iterations diminishing time consumption and memory demands. Finally, the best parameters for our computational capabilities are a field of view of 52×58 pixels and update an 8×8 region in the object mask.

5 Conclusion

SpiralNet is a new method that allows to segment microscopy images of complex shapes with high attention to details. It is a two stage recursive CNN network capable of both semantic and instance segmentation. SpiralNet takes smaller regions of an image and segments the objects inside with high precision, simulating human perception by looking closer to see more details. It could be useful for many medical tasks, where attention to details is required, such as mark melanoma cancer limits to be removed, the reconstruction of neural circuits or check if medicines affect the cell shape, etc.

SpiralNet consists of two stages, the first stage has a scoring network that aims to divide and filter regions without information and outputting only regions with an object located in the center. The second stage has a recursive segmenting network which segments the center-located object. Finally, segmented objects are written to the full final prediction as a one class for the semantic segmentation or as different integers for the instance segmentation task.

SpiralNet predicts well-separated individual objects for instance segmentation and outperformed U-Net for semantic segmentation with F1 score 0.969 on a test set against 0.965 for U-Net, where SpiralNet recall is 0.977 versus 0.961 for U-Net and SpiralNet precision is 0.963 compare to 0.969 for U-Net. Additionally, compared to the ground truth SpiralNet output outlines small details in objects, such as holes or curved edges. This results showcase that SpiralNet is a promising novel method for medical image segmentation.

Although, SpiralNet has shown a good performance, it can be still further improved with hyperparameter tuning. More skip modules could be added to the architecture, parameters like region size and updating step could be changed to boost the network performance even further. Furthermore, the scoring network could be improved to correct errors we have observed. Moreover, the segmentation network can be parallelized, so it segments several separeted objects at the same time speeding up SpiralNet significantly. The future work will study the aplication of SpiralNet to more challenging datasets in comparation with other segmenting methods.

References

[AHY+18] Md. Zahangir Alom, Mahmudul Hasan, Chris Yakopcic, Tarek M. Taha, and Vijayan K. Asari. Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation. *CoRR*, abs/1802.06955, 2018.

[AL19] Feng-Ping An and Zhi-Wen Liu. Medical Image Segmentation Algorithm Based on Feedback Mechanism CNN. 2019:13, 2019.

[blo] Semantic segmentation: Introduction to the deep learning technique behind google pixel's camera! `https://www.analyticsvidhya.com/blog/2019/02/tutorial-semantic-segmentation-google-deeplab/`. Last accessed 26 September 2019.

[CVC+95] L.P. Clarke, R.P. Velthuizen, M.A. Camacho, J.J. Heine, M. Vaidyanathan, L.O. Hall, R.W. Thatcher, and M.L. Silbiger. Mri segmentation: Methods and applications. *Magnetic Resonance Imaging*, 13(3):343 – 368, 1995.

[dat] Broad bioimage benchmark collection. `https://data.broadinstitute.org/bbbc/BBBC039/`. Last accessed 29 April 2019.

[Gir15] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[goo18] Improving connectomics by an order of magnitude. `https://ai.googleblog.com/2018/07/improving-connectomics-by-order-of.html`, 2018. Last accessed 29 April 2019.

[Gra13] Mark L Graber. The incidence of diagnostic error in medicine. *BMJ Quality & Safety*, 22(Suppl 2):ii21–ii27, 2013.

[HGDG17] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[HJHK19] Mohammad Hesam Hesamian, Wenjing Jia, Xiangjian He, and Paul Kennedy. Deep learning techniques for medical image segmentation: Achievements and challenges. *Journal of Digital Imaging*, 32(4):582–596, Aug 2019.

[JML+16] Michal Januszewski, Jeremy Maitin-Shepard, Peter Li, Jörgen Kornfeld, Winfried Denk, and Viren Jain. Flood-filling networks. *CoRR*, abs/1611.00421, 2016.

[KSD+08] Armen R. Kherlopian, Ting Song, Qi Duan, Mathew A. Neimark, Ming J. Po, John K. Gohagan, and Andrew F. Laine. A review of imaging techniques for systems biology. *BMC Systems Biology*, 2(1):74, Aug 2008.

[LGOS13] Shu Liao, Yaozong Gao, Aytekin Oto, and Dinggang Shen. Representation learning: A unified deep learning framework for automatic prostate mr segmentation. In Kensaku Mori, Ichiro Sakuma, Yoshinobu Sato, Christian Barillot, and Nassir Navab, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, pages 254–261, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[LGW+18] Q. Li, Z. Gao, Q. Wang, J. Xia, H. Zhang, H. Zhang, H. Liu, and S. Li. Glioma segmentation with a unified algorithm in multimodal mri images. *IEEE Access*, 6:9543–9553, 2018.

[LL18] Jun Liu and PengFei Li. A mask r-cnn model with improved region proposal network for medical ultrasound image. In De-Shuang Huang, Kang-Hyun Jo, and Xiao-Long Zhang, editors, *Intelligent Computing Theories and Application*, pages 26–33, Cham, 2018. Springer International Publishing.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[LXPS17] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[OSF+18] Ozan Oktay, Jo Schlemper, Loïc Le Folgoc, Matthew C. H. Lee, Mattias P. Heinrich, Kazunari Misawa, Kensaku Mori, Steven G. McDonagh, Nils Y. Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas. *CoRR*, abs/1804.03999, 2018.

[PXP00] Dzung L. Pham, Chenyang Xu, and Jerry L. Prince. Current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, 2(1):315–337, 2000. PMID: 11701515.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C Cortes, N D Lawrence, D D Lee, M Sugiyama, and R Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[RJKK15] Rahimeh Rouhi, Mehdi Jafari, Shohreh Kasaei, and Peiman Keshavarzian. Benign and malignant breast tumors classification based on region growing and cnn segmentation. *Expert Systems with Applications*, 42(3):990 – 1002, 2015.

[ZDL11] K. Zhang, J. Deng, and W. Lu. Segmenting human knee cartilage automatically from multi-contrast mr images using support vector machines and discriminative random fields. In *2011 18th IEEE International Conference on Image Processing*, pages 721–724, Sep. 2011.

[Zha1] Yu-Jin Zhang. Image Segmentation in the Last 40 Years. In *Encyclopedia of Information Science and Technology, Second Edition*, pages 1818–1823. IGI Global, jan 1.

[ZMTL19] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation. *arXiv e-prints*, page arXiv:1912.05074, Dec 2019.

[ZRSTL18] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In Danail Stoyanov, Zeike Taylor, Gustavo Carneiro, Tanveer Syeda-Mahmood, Anne Martel, Lena Maier-Hein, João Manuel R.S. Tavares, Andrew Bradley, João Paulo Papa, Vasileios Belagiannis, Jacinto C. Nascimento, Zhi Lu, Sailesh Conjeti, Mehdi Moradi, Hayit Greenspan, and Anant Madabhushi, editors, *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 3–11, Cham, 2018. Springer International Publishing.

Appendix

I. Code

The source code for SpiralNet method is located in the following GitHub repository:

`https://github.com/anitera/segmentationthesis`

The access to the repository could be granted upon sending an email to:

`marharyta.dekret@gmail.com`

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Marharyta Dekret**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   SpiralNet: Two-stage recursive-CNN for microscopy image segmentation,

   supervised by Daniel Majoral, PhD.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Marharyta Dekret
30/12/2019